Agile Sink Selection in Wireless Sensor Networks

Mohammadreza Mahmudimanesh, Neeraj Suri
Technical University of Darmstadt, Germany
Email: {reza,suri}@cs.tu-darmstadt.de

***Abstract*—The conventional setup of a wireless sensor network is composed of several sensor nodes and one or more sinks. The network topology and data collection techniques are then optimized towards efficient collection of the sensed data at the sink(s). In this paper, we present a novel network coding technique based on Compressed Sensing that allows each node to operate as a sink. Our network coding technique efficiently disseminates a number of linear combinations of the sensed data. After the dissemination phase, the entire sensed data is available by querying any node of the network. This is especially useful for distributed control using a wireless sensor and actuator network or in scenarios where the end user needs to access the global state of the environment from any node in its vicinity; e.g., when the end user is mobile.**

I. Introduction

Wireless Sensor Networks (WSNs) are distributed sensory systems for large-scale monitoring of physical parameters of interest such as seismic vibrations, temperature, light intensity, radiation level, etc. [31]. They have applications in environment monitoring, industrial automation, surveillance, and so on. A WSN consists of battery-powered Sensor Nodes (SNs) that communicate with each other over a wireless medium. Due to energy constraints, the communication range is limited; hence, each SN can only exchange information with a few neighboring nodes [20].

In a typical WSN, all of the sensed data are cooperatively gathered at a dedicated node called *sink* [1]. The end user of the WSN fetches these data from the sink for further processing. In this paper, we present an all-to-all dissemination method such that all of the sensed data are accessible from any node of a WSN. In simple words, each node can potentially be a sink. This grants more flexibility and mobility to the end user of the WSN, since it is possible to access the *global state* of the environment from any arbitrary SN in its vicinity.

A. Problem statement

Consider a WSN consisting of n static SNs. Suppose that vector $f \in \mathbb{R}^n$ is made by stacking the values recorded by the SNs. Thus, the i_{th} entry of vector f, namely f_i, is equal to the value recorded by the i_{th} SN. Throughout this paper, we may use either of the terms spatial *signal* or *vector* interchangeably and both refer to f. The goal of our dissemination method is to make the vector f available to all SNs within a certain time limit. This is useful for Wireless Sensor and Actuator Networks (WSANs) [33] when a distributed control is performed based on the global state of the environment. We also consider the scenario where the global state can be recovered by fetching a limited amount of data from a small subset of the SNs; e.g., when a mobile sink visits some SNs and estimates the global state of the environment by extracting their data.

978-1-4799-4657-0/14/$31.00 © 2014 IEEE

There are two challenges to achieve this goal. First, the number of transmissions by the SNs must be minimized in order to meet bandwidth limitations and also save battery. Second, the dissemination protocol must be light-weight such that it can be easily implemented on the basic hardware platforms of the SNs.

B. Contributions

This paper introduces *Comprensus*, a novel protocol for efficient dissemination of *compressible* data in a WSN. Comprensus draws its concepts from the well known *consensus* methods studied in distributed control literature [25]. Unlike consensus, compressibility of the data plays a decisive role in our protocol. The more compressible the spatial signal is, the less transmissions are required by Comprensus to disseminate the data. Comprensus proves a significant performance gain over consensus techniques by exploiting the compressibility of the sensed data and reducing the number of transmissions.

Our approach has two advantages over the straightforward solution that gathers data at a stationary sink and sends it to the mobile end user. First, our method allows each SN to be a potential sink. Second, there is no need for the end user to be in radio range of the stationary sink. The mobile end user can extract the global state of the environment from any SN in its vicinity. This is especially useful for in-door applications where the base station is not necessarily accessible from any arbitrary location in the environment.

Evaluations show that using Comprensus each SN can have an estimation of the global state with a Signal-to-Noise Ratio (SNR) of more than 50 decibels. This level of SNR indicates that Comprensus can achieve a highly accurate estimation of the global state at all SNs. In Section III we show that under a fixed accuracy requirement a tradeoff between latency and energy consumption can be settled depending on specific application requirements. We investigate scenarios ranging from low-latency energy-aggressive mode to energy-preserving high-latency mode. Comprensus proves to be easily tunable to each of these configurations.

C. Paper organization

Section II briefly reviews the related work. Section II-A gives a preliminary background of the theories that Comprensus is based on. The definitions and theorems described in Section II-A are used in explanation of the Comprensus protocol which is detailed in Section III. Section IV provides evaluation results of the Comprensus protocol applied to a large set of simulated WSNs and compares the performance of

Comprensus to dissemination methods based on decentralized compression via randomized gossiping [29].

II. Background and related work

Several studies report that spatial signals recorded by dense WSNs are compressible under a linear transform such as Discrete Cosine Transform (DCT) or Discrete Wavelet Transform (DWT) [11], [19], [21]–[23]. This means that vector f can be written as $f = \Psi x$ for some orthonormal matrix Ψ such that x is almost sparse. More precisely, except a few significant components, all other components of x are negligibly small.

The challenge of applying transform compression to WSNs is that one does not know the location of the significant components of x ahead of time. Therefore, it is hard to implement these techniques in a distributed manner for WSNs [12]. The theory of Compressed Sensing (CS) solves this problem by introducing a novel sampling technique to recover f from a few random linear measurements [7], [8]. Its simple sampling technique and minimum coordination overhead as well as robustness to noise makes it particularly suitable for implementation in WSNs [10]. We will explain the basics of the CS theory in more details in Section II-A.

Various adaptations of the CS theory are proposed for distributed sensing in WSNs to handle the problem of collecting the data at a single sink [2], [3], [15], [19], [22]. These techniques are discussed in more details in Section II-B1 after explaining the preliminary background of the CS theory. Our work is closely related to [29]. Rabbat et al. in [29] employ average consensus via randomized gossiping [5] to disseminate the measurements in a WSN. The novelty of Comprensus is its efficient network coding technique. We show that Comprensus requires much less iterations and transmissions than [29] to disseminate the measurements. We will revisit data dissemination based on compression via randomized gossiping [29] in Section II-B2 after describing the basics of the CS theory required for the explanation of this technique. We show that Comprensus significantly outperforms compression via randomized gossiping [5] both in terms of timely dissemination and reducing the number of in-network transmissions.

Next, we briefly review the fundamentals of the CS theory and study its applications in WSNs.

A. *Compressed Sensing*

The CS theory is initially based on the Restricted Isometry Property (RIP) [9]. This paper is based on the newer version of the CS theory that does not require the RIP. The so called *RIPless* CS theory [6] allows for a computationally feasible method to certify whether the preconditions of accurate signal recovery hold for a particular setup of Comprensus.

We call a vector $a \in \mathbb{R}^n$ a *sensing vector*, and the inner product of a sensing vector and vector f is called a *measurement*. Let $y_1, y_2, \ldots, y_m$ be m measurements such that

$$y_j = {a_j}^T f + \sigma z_j \quad , \quad j \in \{1, 2, \ldots, m\} \tag{1}$$

where a_j are the sensing vectors, $\{z_j\}$ is the white noise sequence and σ^2 is the noise variance. This can be also written using matrix notations:

$$y = Af + \sigma z \tag{2}$$

where $y = [y_1 \ y_2 \ \ldots \ y_m]^T$, $A = [a_1 \ a_2 \ \ldots \ a_m]^T$ and $z = [z_1 \ z_2 \ \ldots \ z_m]^T$.

CS allows to recover the n-dimensional vector f from $m < n$ measurements under certain conditions for f and the sensing vectors as follows. We assume that f can be sparsely represented under a linear projection using an orthonormal matrix Ψ. Suppose that $f = \Psi x$ for a suitably chosen orthonormal matrix Ψ such that x is sparse. Vector x is called a sparse vector when it has $s \ll n$ non-zero components and all its other $(n - s)$ components are zero. Sparsity plays an important role in the CS theory. The sparser the vector x is, the fewer measurements are required to recover f. The spatial signals recorded by a WSN admit a nearly sparse representation under an orthonormal linear transform such as Fourier, DCT or DWT [23].

1) Isotropy and Incoherence: Isotropy and *incoherence* are the other necessary conditions in order to recover f from y [6]. Let $a \in \mathbb{R}^n$ be a random sensing vector with independent and identically distributed components drawn from distribution F, i.e., $a \overset{iid}{\sim} F$.

Definition 1. [6] *Distribution F has the* isotropy property*, when aa^T is expected to be the identity matrix. Mathematically,*

$$E[aa^T] = I \quad , \quad a \sim F. \tag{3}$$

The isotropy condition can be weakened to *near isotropy*, i.e., $E[aa^T] \approx I$ and still f is accurately recoverable from the measurement vector y [6].

Definition 2. [6] Coherence parameter μ *is defined as the smallest value μ such that*

$$|a_j^T \psi_i|^2 \ \leq \ \mu \tag{4}$$

for all sensing vectors a_j and columns ψ_i of Ψ, $j \in \{1, 2, \ldots, m\}$ and $i \in \{1, 2, \ldots, n\}$.

We say that the sensing vectors are more *incoherent* if the value of μ is a smaller number. According to [6], the more incoherent the sensing vectors are, the less random measurements are required for accurate recovery. Candes et al. in their RIPless theory of CS [6] discuss some of the random distributions F obeying the isotropy condition. These include the Gaussian distribution, Rademacher distribution and random Fourier sampling [6]. Randomized sampling brings a key benefit for WSNs by eliminating the need for centralized coordination [3], [19].

It is shown in [6] that if the isotropy condition holds and the number of measurements m is in the order of $O(\mu s \log n)$, then f can be recovered from y with an overwhelming probability. Therefore, we need a basis[1] Ψ and a set of sensing vectors with isotropy property such that f is compressible under Ψ and the columns of Ψ have the least coherence with the sensing vectors. In this paper, we propose a novel network coding technique that fulfills these conditions.

[1] A basis for $\mathbb{R}^n$ is a set of vectors $\psi_i, i \in \{1, \ldots, n\}$, such that any vector $f \in \mathbb{R}^n$ can be represented as $f = \sum_{i=1}^{n} x_i \psi_i$ where x_i are called the coefficients of f in basis Ψ. The corresponding transformation matrix Ψ is made by putting the vectors ψ_i in a matrix, i.e., $\Psi = [\psi_1 \ \psi_2 \ \ldots \ \psi_n]$ and $f = \Psi x$ where $x = [x_1 \ x_2 \ \ldots \ x_n]^T$.

2) Signal recovery: In order to recover f from y, first we need to solve the following convex optimization problem [6].

$$\underset{\bar{x}\in\mathbb{R}^n}{\text{minimize}}\ \frac{1}{2}\|A\Psi\bar{x}-y\|_2^2+\lambda\sigma\|\bar{x}\|_1 \tag{5}$$

where $\lambda = 10\sqrt{m\log n}$, $\|\cdot\|_1$ is the norm-1 operator and $\|\cdot\|_2$ is the norm-2 operator[2].

Some of the efficient and accurate algorithms for solving this problem can be found in [4], [18]. If $\hat{x}$ is the solution to the convex optimization problem in Equation 5, then $\hat{f} = \Psi\hat{x}$ will estimate the original signal f with an error bounded by $\text{polylog}(n)(s/m)\sigma^2$ [6]. In practice, signal f is not strictly sparse under the Ψ-transform. Instead, x has a few components with larger magnitudes and most of its remaining components are nearly zero. Suppose that x_s is a sparse approximation of x by keeping s largest components of x and zeroing its remaining $(n-s)$ components. It is shown in [6] and [8] that, in such a case the recovery error will not grow much more than $O(\|x-x_s\|_2)$.

B. Applications of CS in WSNs

Suppose that each SN of a WSN senses one entry of the spatial signal vector f. The i_{th} entry of f, namely f_i, is equal to the value sensed by the i_{th} SN. Suppose that SN i randomly generates the i_{th} column of matrix A. Let α_i be the i_{th} column of A. Then, $y = f_1\alpha_1 + f_2\alpha_2 + \cdots + f_n\alpha_n$ can be calculated through aggregation over a spanning tree formed on the WSN and the measurement vector y is delivered to the sink [19]. Efficient protocols for data fusion and aggregation using spanning trees is studied in [32].

1) Data collection: We assume that each SN is given a unique id and runs a pseudo-random number generator algorithm seeded by its id to produce α_i. All of the SNs run the same pseudo-random number generator algorithm, though with different seeds. The pseudo-random number generator algorithm should be selected such that it is unlikely that two SNs generate the same sequence of random values. The measurement matrix A can be easily reproduced at the sink by executing the same pseudo-random number generator algorithm seeded by every SN id. Therefore, the measurement matrix A does not need to be communicated between the SNs and the sink. The SNs and the sink only need to agree on a common pseudo-random number generator algorithm. Having A and y, the sink can recover f from y as detailed in Section II-A2. This process forms the main building block of many distributed data gathering techniques based on CS [2], [15], [19], [22].

2) Data dissemination using randomized gossiping: Dissemination of the random measurements using average consensus algorithm was first studied in [29]. The measurement mechanism of [29] is similar to what we explained above for data gathering. The difference is that a gossiping average consensus technique is applied to disseminate the measurements $y_j, j \in \{1,\ldots,m\}$ in the network.

Suppose that SN i computes $f_i\alpha_i$ and stores the result in an array of m real numbers. Let $w_i[t] \in \mathbb{R}^m$ refer to the content of the array inside SN i at time t. SN l_1 is activated uniformly at random at time t and chooses one of its neighbors l_2 uniformly at random. SN l_1 and SN l_2 exchange $w_{l_1}[t]$ and $w_{l_2}[t]$ and update $w_{l_1}[t+1] = w_{l_2}[t+1] = (w_{l_1}[t] + w_{l_2}[t])/2$. When $t \to \infty$, $w_i \to Af$ for all $i \in \{1,2,\ldots,n\}$ [29]. Therefore, after *sufficiently many* iterations of this protocol w_i in all SNs will *get close enough* to $y = Af$, and the signal f can be recovered at any SN after solving Equation 5. Since the size of y and also w_i is in the order of $\mu s\log n$, randomized gossiping requires $O(\mu s\log n)$ transmissions per iteration. The number of required iterations depends on the network topology [29].

One drawback of this method is that the measurements are inaccurate unless sufficiently many iterations of this protocol are executed. In practice, too many iterations and message exchanges are required to achieve the consensus below an acceptable error threshold. This paper proposes a novel network coding mechanism which is still as simple as consensus, nevertheless, requires less time and communications to disseminate the measurements. In [29], the SNs run a protocol such that all of them *converge* to a measurement vector y which is common among all SNs. In Comprensus, each SN receives a different measurement vector y_i, $i \in \{1,2,\ldots,n\}$. We show that for all of these measurement vectors, the isotropy and incoherence properties hold. Therefore, Comprensus does not need too many iterations for convergence to the same measurement vector among all SNs. Instead, our proposed method guarantees that the same signal f is accurately recoverable from each individual measurement vector y_i received by SN i, $i \in \{1,2,\ldots,n\}$.

III. The Comprensus protocol

In this section, we explain Comprensus, a simple distributed protocol to disseminate random linear measurements in a WSN with static topology. We assume that the network topology corresponds to a connected regular graph of degree d. It is easy to create a regular graph topology in a WSN when nd is even. For a given degree d each SN selects at most d neighbors with the highest Received Signal Strength Indicator (RSSI) [30] assuming that each SN has at least d SNs in its communication range. At the end of this process, we will have a topology corresponding to a regular graph of degree d.

We start by defining the variables and definitions used in our protocol. Suppose that each SN is equipped with two pseudo-random number generators as defined below.

- *Rademacher random generator* produces either $+1$ or -1 each with probability $1/2$.
- *Bernoulli random generator* produces 1 with probability $p = k/n$ and 0 with probability $1-p$.

We assume that SN i keeps a real number u_i in its internal memory.SN i also keeps a list L_i of real numbers in its memory. The data type of the elements of L_i is the same as the data type of u_i. Memory requirement for this list is $O(\mu s\log n)$ items. We will show shortly that μ will be a small constant. This list actually holds the random linear measurements which are used thereafter for signal recovery. One can have an estimation of s in an appropriate basis Ψ based on a previous knowledge about the data gathered from the WSN. Since this estimation is not necessarily accurate,

[2]For a real vector $v \in \mathbb{R}^n$, norm-1 of v is defined as $\|v\|_1 = \sum_{i=1}^n |v_i|$ and norm-2 of v is defined as $\|v\|_2 = \sqrt{\sum_{i=1}^n |v_i|^2}$

it is recommended to use a worst case estimation for s in a real-world deployment of Comprensus.

A. Distributed Comprensus algorithm

The Comprensus protocol is executed in three phases: *Initialization*, *Dissemination* and *Recovery*. The instructions described below will be executed in parallel by every SN i, $i \in \{1, 2, \ldots, n\}$.

1) Initialization: First, the list L_i is emptied. Then, SN i reads the value f_i from its sensor and stores it into variable u_i. We assume that each SN is given a unique id and initializes the seeds of the Rademacher and Bernoulli random generators by its id. By choosing an efficient and reliable pseudo-random number generator we minimize the chance that two SNs generate the same sequence of random values [24].

2) Dissemination: This phase is repeated r times in parallel by all n SNs. At each iteration $t \in \{1, 2, \ldots, r\}$ all of the SNs execute Algorithm 1 simultaneously.

- $h_i[t]$ is the value generated by the Rademacher random generator of SN i at iteration t.
- $b_i[t]$ is the value generated by the Bernoulli random generator of SN i at iteration t.

Algorithm 1 Dissemination phase of Comprensus

```
1: u_i ← h_i[t] · u_i
2: if b_i[t] = 1 then
3:     Transmit u_i
4: else
5:     for all SN j in neighborhood of SN i do
6:         if SN j is transmitting the value u_j[t] then
7:             u_i ← u_i + u_j[t]/n
8:         end if
9:     end for
10: end if
11: if at least one neighbor has transmitted then
12:     add u_i to the rear of L_i
13: end if
```

Remark: if a set of adjacent SNs want to transmit at the same iteration, they transmit one by one according to the descending order of their ids. They aggregate their received measurements in a temporary variable and update their corresponding u_i only after all of these concurrent transmissions are completed. If the SNs are perfectly synchronized, aggregation by signal superposition helps to perform this step *simultaneously* by all of these adjacent SNs [2], [3].

3) Recovery: SN i derives a vector y_i by stacking the entries in list L_i. When all SNs agree on a common random generator algorithm, the linear combinations that led to the values in L_i are reproducible as described in Section II-B1. These linear measurements are then placed in Equation 5 to recover f. We show in Section III-C that the linear measurements acquired in the dissemination phase obey the isotropy condition and have low coherence with DCT.

Line 1 of Algorithm 1 generates a new Rademacher value and multiplies it by the current value of u_i which is first set to f_i in the initialization phase. Line 2 decides whether SN i is to transmit in this iteration or not. Since $b_i[t]$ returns 1 with probability k/n, this is equivalent to the case that almost k out of n SNs select themselves uniformly at random to transmit. Executing the line 3 consumes the most amount of battery power, as using the radio in transmitting mode is the major energy drain of a SN [20]. If SN i is not in transmitting mode at iteration t, i.e., $b_i[t] = 0$, then it listens to the communication channel and accumulates the values sent by neighboring nodes onto u_i after dividing them by n as instructed in lines 5 through 9. Summing the received values from neighboring nodes can be done arithmetically by using a simple Time Division Multiple Access (TDMA) mechanism [1]. A faster alternative is signal superposition as proposed in [2], [3] when the SNs are perfectly synchronized. It can also happen that no neighbor of SN i does a transmission at iteration t. In this case, no value is added to the list L_i. This condition is checked in line 11, and thus, line 12 is executed only when at least one neighboring node has transmitted. We will explain shortly why this restriction is necessary.

B. Matrix representation of the distributed protocol

In this section we examine the network-wide implication of Algorithm 1 by using the equivalent matrix representation of Comprensus.

Let N_i denote the set of the d neighbors of SN i.

Definition 3. *Transition matrix M_t at iteration t is an $n \times n$ real matrix with the following attributes.*

1) *$M_t[i, i] = h_i[t]$ for $1 \le i \le n$.*
2) *$M_t[j, i] = h_i[t]/n$ when $j \in N_i$ and $b_i[t] = 1$.*

It is easy to verify that after iteration t of the dissemination phase,

$$\begin{pmatrix} u_1[t] \\ u_2[t] \\ \vdots \\ u_n[t] \end{pmatrix} = (M_t \times M_{t-1} \times \cdots \times M_1)\, f \tag{6}$$

describes the contents of variables u_i , $i \in \{1, 2, \ldots, n\}$. We also define the $n \times n$ matrix Q_t as

$$Q_t := \begin{pmatrix} q_{1,t} \\ q_{2,t} \\ \vdots \\ q_{n,t} \end{pmatrix} := M_t \times M_{t-1} \times \cdots \times M_1 \tag{7}$$

where $q_{1,t},\ q_{2,t}\ ,\ \ldots\ ,\ q_{n,t}$ are the rows of matrix Q_t.

We define a set R_i as

$$R_i := \{t \quad | \quad \exists j \in N_i \ : b_j[t] = 1\} \tag{8}$$

to refer to the set of iterations in which at least one neighboring node of SN i is transmitting. We also define matrix $\mathcal{A}_i$ as

$$\mathcal{A}_i := [q_{i,t_1}^T \ q_{i,t_2}^T \ \ldots \ q_{i,t_{m(i)}}^T]^T \tag{9}$$

where $m(i) = |R_i|$ is the number of measurements received by SN i and $\{t_1, t_2, \ldots, t_{m(i)}\} = R_i$. The number of received measurements may differ from one SN to other. Nevertheless, when the network topology corresponds to a regular graph, all of the nodes are expected to receive almost the same amount of measurements, since each SN has an equal chance to transmit and receive messages. It can be shown that the measurement vector y_i made by stacking the values in list L_i will be

$$y_i = \mathcal{A}_i f + z \tag{10}$$

where z is the additive noise. The noise is added either by the communication channel or can be regarded as a side effect of low precision floating pointing storage and processing inside the SNs. We model z by a white Gaussian noise vector in our simulations and experiments.

If the rows of $\mathcal{A}_i$ obey the isotropy property and have low coherence with a compressive basis, then f can be recovered at SN i from y_i as detailed in Section II-A2. Now the reason for the restriction in Line 11 of Algorithm 1 becomes clear. We only let *newly received* measurements to be aggregated and added to the measurement list L_i. Otherwise, $\mathcal{A}_i$ will have at least two rows which are linearly dependent, and thus, $\mathcal{A}_i$ is not full rank. In other words, we will have redundant measurements stored in L_i if we do not check the condition in Line 11 of Algorithm 1.

Suppose that $\bar{m}$ is the average number of measurements received per SN. $\bar{m}$ should be in order of $O(\mu s \log n)$ in order to perform successful recovery. When these conditions are fulfilled, the signal vector f can be recovered at every SN after running the Comprensus protocol. Next, we examine isotropy and incoherence properties of our measurement matrix $\mathcal{A}_i$ for $i \in \{1, 2, \ldots, n\}$.

C. Numerical experiments

In this section, we investigate the isotropy and incoherence of our measurement method through numerical experiments on simulated WSNs. We perform comprehensive numerical experiments on simulated WSNs consisting of $n = 100$ SNs. Note that even better performance is expected for larger n because of the logarithmic cost growth with n. The network graph is a random regular graph of degree $d = 5$ which is freshly generated in each experiment and the results are averaged over multiple runs. We let the SNs to generate their corresponding h_i and b_i random numbers and execute the Comprensus protocol for varying values of r and k. Each experiment is run 100 times and all of the results are averaged to eliminate randomness effects.

In Section II-A1, we have seen that even if the set of sensing vectors have the *near-isotropy* property, the signal f can be recovered from measurement vector y. In Comprensus, the set of the sensing vectors for SN i are the rows of $\mathcal{A}_i$ and the measurement vector for SN i is y_i. We define a metric for *deviation from isotropy* and show that the rows of $\mathcal{A}_i$ have a very low deviation from the isotropy property.

Definition 4. Deviation from isotropy *for a random sensing vector a is defined as $\sum_{e \in \mathcal{E}_a} (1-e)^2$ where $\mathcal{E}_a$ is the the set of eigenvalues of the square matrix $E[aa^T]$.*

This metric determines how much $E[aa^T]$ behaves like an identity matrix. In the ideal case, $E[aa^T] = I$ and has only one eigenvalue, i.e., 1, and thus, the deviation from isotropy is zero. For random sensing vector a, if the eigenvalues of $E[aa^T]$ are all very close to 1, $E[aa^T]$ behaves like an identity matrix and deviation from isotropy as defined in Definition 4 will be low.

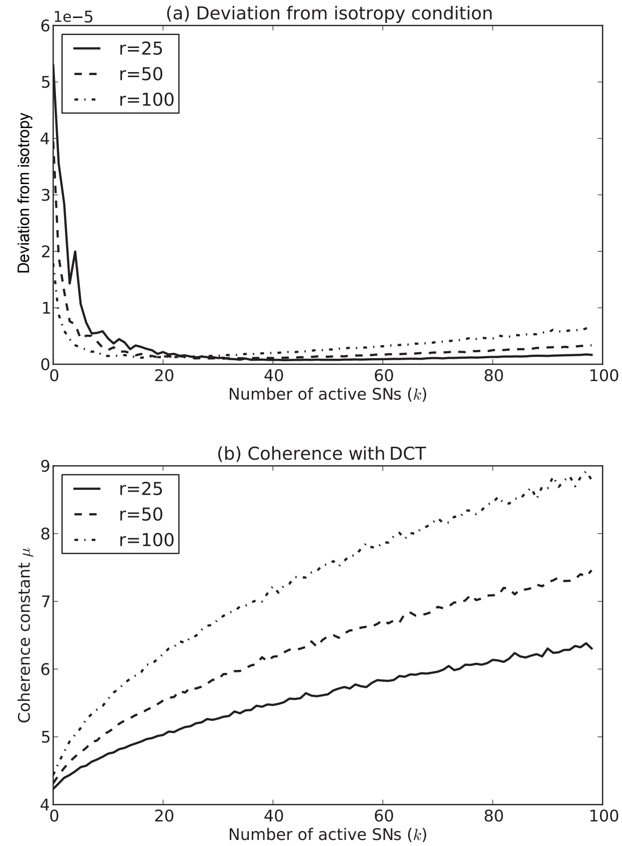


Fig. 1. Near-isotropy and low coherence of Comprensus measurements

In our numerical experiments, a large set of measurement matrices $\mathcal{A}_i$ are generated. Our random sensing vectors are actually the rows of the randomly generated measurement matrices $\mathcal{A}_i$. For each row a^T of these matrices, we calculate aa^T and sum up all of the results. $E[aa^T]$ is then numerically calculated by dividing this summation by the total number of the randomly generated sensing vectors.

Observation 1 – near-isotropy: Deviation from isotropy according to Definition 4 is calculated over all randomly generated measurement matrices $\mathcal{A}_i$, $i \in \{1, 2, \ldots, n\}$ and a full range of experiments with k varying from 1 to n and $r \in \{25, 50, 100\}$. The results as illustrated in Figure 1(a) prove that our measurement mechanism obeys the near-isotropy property with negligible deviation, i.e., $E[aa^T] \approx I$.

Observation 2 – low coherence: We set Ψ to be the inverse DCT matrix and calculate the coherence with the DCT basis according to Definition 2 for randomly generated measurement matrices with k varying from 1 to n and $r \in \{25, 50, 100\}$. The averaged results over 100 simulations as illustrated in Figure 1(b) shows that the coherence factor is also low.

D. Dissemination and recovery strategies

Dissemination and recovery are the two building blocks of Comprensus which are implemented in a decoupled manner.

Here, we discuss why this decoupling allows a more flexible and customizable protocol for data dissemination in WSNs. Depending on the processing capabilities of the SNs, we follow one of the following strategies.

- *Recovery at SNs*: Individual SNs have enough processing power to run the recovery algorithm on their received measurement vectors y_i and access f.
- *Recovery at an external collector*: An external data collector that has enough processing power retrieves y_i from any SN in its vicinity and recovers f.

Note that, in the second case, although the SNs themselves can not extract the global vector f, their measurement vectors contain enough information about the global spatial signal f. Not being able to extract the global state is a consequence of their low processing power. In other words, the information is contained in their perceived measurement vectors. Nevertheless, they have not sufficient processing power to run the reconstruction algorithm and extract the original data. [27] presents a high-performance SN platform that can run Comprensus with the first strategy. Telos [28] is a widely used SN platform that is suitable for the second strategy.

Isotropy and incoherence are the crucial prerequisites for Comprensus to work. At the same time, the average number of measurements $\bar{m}$ that are received at each SN must be in order of $O(\mu s \log n)$ to recover f. Figure 2(a) depicts how $\bar{m}$ increases with k for $r \in \{25, 50, 100\}$. Signal recovery accuracy of Comprensus improves when μ is low and $\bar{m}$ is high. Figure 2(b) shows how $\bar{m}/\mu$ changes with the given simulation parameters. One interesting observation is that, there exists a maximum ratio of $\bar{m}/\mu$ corresponding to a suitably chosen k for a given r. Effectiveness of Comprensus is directly related to $\bar{m}/\mu$. The most desirable situation is when the user requirement exactly matches the configuration with maximum $\bar{m}/\mu$ for $\bar{m}$ being high enough for signal recovery. However, the user may have different demands, for example minimizing the energy consumption to prolong the lifetime of the WSN might be a higher priority in certain applications. Most energy-preserving configuration is attainable when rk is minimized while $\bar{m}$ is high enough to recover f. We will further discuss these scenarios in Section IV where we investigate the performance of the Comprensus protocol.

Comprensus can be also extended to a system with static SNs and mobile sinks or multiple sinks that can retrieve data from a small subset of SNs. Suppose that an external collector polls a subset $P \subset \{1, 2, \ldots, n\}$ of the SNs and retrieves their measurement vectors y_j, $j \in P$. The most effective configuration of Comprensus will be selecting r and k such that $\bar{m}|P|/\mu$ is maximized. $|P|$ is in fact the number of SNs that the collector can poll. Again a tradeoff between r, k and the number of polled SNs can be settled to maximize the effectiveness of Comprensus.

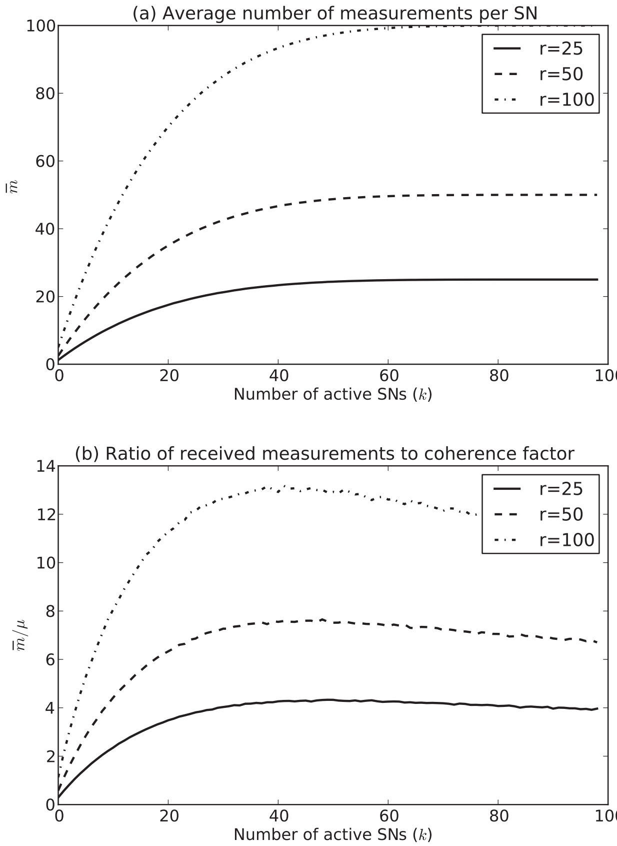


Fig. 2. Effectiveness of Comprensus depending on the average perception per SN

IV. Evaluation

In this section, we investigate how accurate the signal f can be recovered at an arbitrarily chosen SN i from its measurement vector y_i. We compare Comprensus with another state of the art method based on compression via randomized gossiping [29]. We evaluate a large number of simulated WSNs consisting of $n = 100$ SNs. The network topology is a regular graph of degree 5. We assume that our desired topology is accomplished by an appropriate topology control mechanism [30]. Evaluations are performed by numerical experiments in SciPy [16]. Each run of the experiment is conducted for a freshly-generated random regular graph to ensure that the results are valid for any arbitrary network with balanced topology.

The compressible signal f is constructed by calculating the inverse DCT transform of a sparse random vector that has s nonzero components drawn from a normal distribution. This way, we have a signal vector f whose projection on DCT has only s nonzero coefficient. We conduct the experiment for $s = 5$, $s = 10$ and $s = 20$ to discover the effect of sparsity on the recovery performance. We run the Comprensus protocol for $1 \leq k \leq n$ and $1 \leq r \leq n$. After running the protocol, SN i retrieves the measurement vector y_i from L_i. We use CVXOPT [13] to recover the original signal f from y_i by using our implementation of the convex optimization solver to find the solution of Equation 5. In every run, we pick a SN at random to perform the recovery on its measurement vector. This random selection ensures that it is possible to recover f at *any arbitrarily chosen SN*. The accuracy of signal reconstruction is measured by SNR in decibels (dB).

A. Signal recovery performance

Figure 3 illustrates the result of our experiments for three different compressibility levels of f. Figures 3(a), 3(b) and 3(c) show the SNR of the signal recovery, when projection of f on the DCT basis has 5, 10, and 20 coefficients respectively. The bright area indicates high SNR in signal reconstruction. This essentially means that accurate dissemination is possible for a particular r and k when the point at position (r, k) falls in the bright area of the SNR diagrams of Figure 3. We observe that, as s increases, we require either more transmissions (k) per iteration or more iterations (r) in the dissemination phase of Comprensus. This is reflected in Figure 3 as the bright area of the SNR diagram shrinks when s increases.

1) Energy consumption reductions: The dissemination phase of the Comprensus protocol is executed in r iterations and in each iteration, k SNs are transmitting. Therefore, the energy consumption is in order of $O(rk)$ for a full dissemination, since radio transmission is the dominant energy drain of a SN [20]. We will see shortly that the communication complexity is much less than randomized gossiping methods which require $\Omega(ns \log n)$ best case (full connectivity of all nodes, i.e., network topology corresponding to a complete graph) and $\Theta(n^2 s \log n)$ worst case [26], [29]. Also note that none of these state of the art methods provide the tunability that Comprensus offers. Using Comprensus, the user can easily tune the dissemination protocol to operate either fast in expense of more energy consumption or slower for the sake of battery saving.

Comprensus requires at most $O(rd)$ computations per SN during the dissemination phase. Our evaluations presented in the next section show that for $r = n$ accurate signal recovery is doable. By setting d to a small integer we will have $O(n)$ computation complexity per SN in the dissemination phase of Comprensus. The actual processing complexity in the recovery phase of Comprensus depends on whether signal reconstruction takes place on the SN or on an external node as described in Section III-D. As the embedded microprocessors are becoming faster and more energy-efficient, we can expect that the recovery algorithm can be executed on contemporary higher-end mobile or embedded processors. One interesting example is an electrocardiography monitor introduced in [17] that runs the CS recovery algorithm on an iPhone. Another possibility is that a dedicated node polls some of the SNs and retrieves their measurement vectors y_i. The recovery algorithm is then executed on this dedicated node which has sufficient processing power. One example can be a mobile sink or a WSN with multiple dedicated sinks. The more SNs that can be efficiently polled in this method, the less messages need to be exchanged during the dissemination phase of Comprensus.

2) Dissemination latency reductions: In a low-latency regime, it is required to disseminate the data in the least amount of time. Looking at the three diagrams of Figure 3, we realize that the minimum r is obtained when k is maximized, i.e., all SNs transmit their $u_i[t]$ at all iterations $t \in \{1, 2, \ldots, n\}$. This is the fastest and most energy-consuming mode of Comprensus' dissemination mechanism. On the other hand, we observe that beyond a certain value of k, the minimum number of iterations required for accurate recovery does not grow significantly. In Figure 3(a), we see that the point ($r = 30$, $k = 40$) falls on the bright area, and hence, accurate signal recovery is possible for $r = 30$ and $k = 40$. When we double the parameter k, the minimum number of iterations r required to stay in the bright area (recoverable condition) only decreases by at most 5 iterations. The same pattern is seen in Figure 3(b) and Figure 3(c) as the curve dividing the dark and the bright areas becomes almost vertical for larger k. Therefore, although the minimum number of required iterations r decreases by increasing the number of active nodes k, the *growth* of performance gain drops drastically for larger values of k.

3) Sharp phase transitions: An interesting observation in our evaluations is that the transition to the condition where accurate recovery is possible is relatively sharp. Looking at the SNR diagrams of Figure 3, the border between the bright area (successful dissemination) and dark area (non-recoverability) has a recognizable contrast. Sharp transition between recoverability and non-recoverability states in CS is comprehensively studied in the Donoho-Tanner universal phase transition inspections [14].

B. Comparison to randomized gossiping methods

In this section, we compare Comprensus to decentralized compression based on randomized gossiping [29]. We set our SNR requirement for both protocols to 40 dB and compare their performance in achieving this requirement. Network topology and all other conditions are also the same for both protocols.

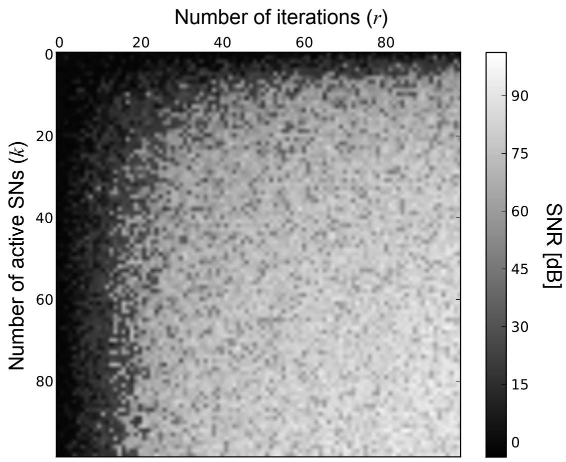
As described in Section II-B2, randomized gossiping requires at least $s \log n$ data exchanges per iteration. The number of required iterations is dependent on the network topology and maximum allowed measurement error. Assume that $w_i[t]$ is the content of the measurement vector of SN i at iteration t of randomized gossiping as defined in Section II-B2. As we have seen in Section II-B2, $\|w_i[t] - y\|_2 \rightarrow 0$ for $y = Af$ when $t \rightarrow \infty$. We define the measurement error of SN i at iteration t as

$$\epsilon_{i,t} \ := \ \|w_i[t] - y\|_2. \tag{11}$$

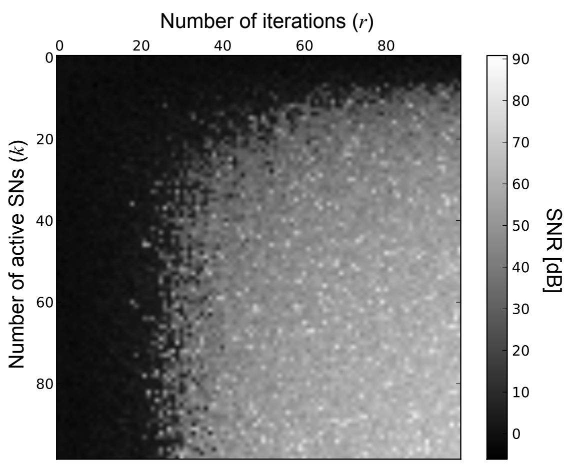
We also define average measurement error at iteration t as $\epsilon_t := (\sum_{i=1}^{n} \epsilon_{i,t})/n$. From the arguments in Section II-A2 we know that for recovering f with SNR of at least 40 dB, the measurement error $\epsilon_{i,t}$ must be lower than $\|f\|_2 \times 10^{-4}$ for the measurement vector received by SN i at time t.

Figure 4 shows how ϵ_t decays with the number of randomized gossiping iterations for a WSN consisting of $n = 100$ SNs with a network topology corresponding to a connected regular graph of degree $d = 5$. We observe that, average measurement error goes below our required threshold after almost 1200 iterations. We round down this number to 1000 iterations in favor of the randomized gossiping method.

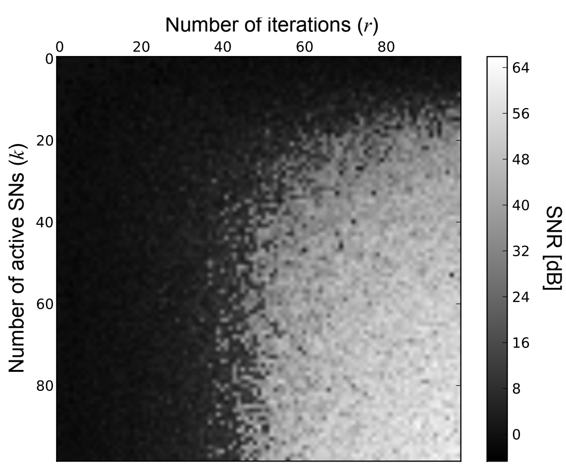
Now we compare the total amount of transmissions in Comprensus and randomized gossiping. We consider the three test cases illustrated in Figure 3. For $s \in \{5, 10, 20\}$, randomized gossiping requires $s \log n \ \approx \ 4.6 \times s$ transmissions per iteration when $n = 100$, and thus, almost $4.6 \times 10^3 \times s$ transmissions in total, since it requires to execute 1000 iterations. Comprensus needs rk transmission corresponding to (r, k) that



(a) $s = 5$



(b) $s = 10$



(c) $s = 20$

Fig. 3. Accuracy of signal recovery for different runs of Comprensus dissemination protocols

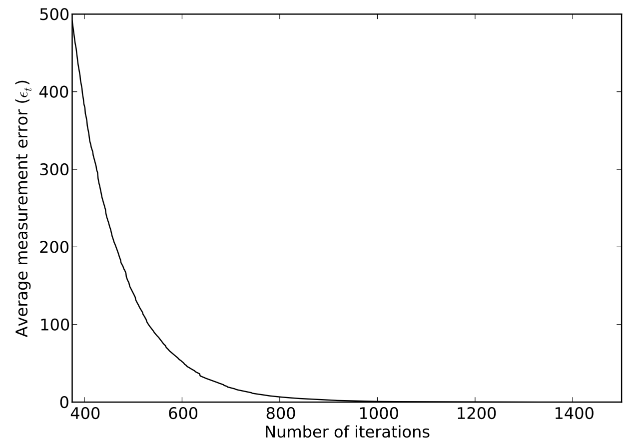


Fig. 4. Measurement error decay with iterations of randomized gossiping

falls on the bright part of the SNR diagram of Figure 3. For s being 5, 10 and 20 we set $(r = 50, k = 30)$, $(r = 60, k = 40)$ and $(r = 80, k = 40)$ respectively. Looking at Figure 3, we see that these are rather conservative selections and signal recovery is possible with fewer numbers of transmissions. Nevertheless, we run Comprensus with these conservative settings and compare its performance to the randomized gossiping method. The comparison result is summarized in Table I.

TABLE I. COMPARING COMPRENSUS TO RANDOMIZED GOSSIPING

s	Total number of transmissions	
	Comprensus	Randomized gossiping
5	1.5×10^3	2.3×10^4
10	2.4×10^3	4.6×10^4
20	3.2×10^3	9.6×10^4

Comprensus proves to disseminate the random linear measurements not only in significantly less number of iterations, but also using much less amount of in-network transmissions. Using its efficient network coding technique, Comprensus disseminates compressible data with low latency and high quality while keeping the number of transmissions as low as possible in order to preserve more battery power of the SNs.

V. Conclusion

In this paper, we introduced Comprensus, a light-weight and efficient protocol for the dissemination of the sensed data in a wireless sensor network. Comprensus allows each sensor node to access the global state of the environment. Availability of the global state of the environment at all sensor nodes allows for a more flexible deployment of wireless sensor networks, for example when a mobile sink wants to access all of the sensed data by fetching data from any sensor node in its vicinity.

Comprensus is based on the recent advances in the compressed sensing theory for the recovery of compressible data from a limited number of random linear measurements. We know that the spatial signals recorded by wireless sensor networks admit highly compressible representations under an appropriate linear transform such as Fourier, discrete cosine transform or discrete wavelet transform. Comprensus exploits this compressibility and reduces the number of in-network transmissions required for the dissemination of the sensed data.

Our proposed protocol employs a novel network coding mechanism for the dissemination of the random linear measurements that comply with isotropy and incoherence properties as required by the compressed sensing theory. The original spatial signal is then recovered from these random linear measurements by running a reconstruction algorithm. Depending on the hardware capabilities of the sensor nodes we follow either of these strategies: the signal recovery is fulfilled on the sensor nodes, or it can be offloaded to an external data collector. We have discussed both scenarios and analyzed their corresponding communication and computation complexities.

Apart from efficient measurement and dissemination mechanisms, Comprensus allows a high level of flexibility and tunability according to specific application requirements. We have investigated scenarios ranging from fast and energy-aggressive data dissemination to slower and energy-preserving dissemination of the sensed data. We have shown that our protocol is easily adaptable to each of these requirements.

As a future work, we are going to extend Comprensus to wireless sensor networks with dynamic topologies or mobile sensor nodes.

Acknowledgment

The authors are partly supported by the Research Training Group (Graduiertenkolleg) 1362 of the German Research Foundation (Deutsche Forschungsgemeinschaft - DFG).

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393 – 422, 2002.

[2] W. Bajwa et al. Compressive wireless sensing. In *Proc. of the 5th international conference on Information processing in sensor networks (IPSN)*, pages 134–142, 2006.

[3] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak. Joint source-channel communication for distributed estimation in sensor networks. *IEEE Trans. on Information Theory*, 53(10):3629–3653, 2007.

[4] T. Blumensath and M. E. Davies. Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis*, 27(3):265–274, 2009.

[5] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Gossip algorithms: design, analysis and applications. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 3, pages 1653–1664 vol. 3, 2005.

[6] E. Candes and Y. Plan. A probabilistic and ripless theory of compressed sensing. *IEEE Tran. on Information Theory*, 57(11):7235–7254, 2011.

[7] E. Candes and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Trans. on Information Theory*, 52(12):5406 –5425, 2006.

[8] E. J. Candes. Compressive sampling. In *Proc. of the International Congress of Mathematicians*, volume 3, pages 1433–1452, 2006.

[9] E. J. Candes. The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique*, 346(910):589 – 592, 2008.

[10] Z. Charbiwala et al. Compressive oversampling for robust data transmission in sensor networks. In *IEEE INFOCOM'10*, pages 1 –9, 2010.

[11] A. Ciancio et al. Energy-efficient data representation and routing for wireless sensor networks based on a distributed wavelet compression algorithm. In *ACM IPSN'06*, pages 309–316, 2006.

[12] R. Cristescu et al. Network correlated data gathering with explicit communication: Np-completeness and algorithms. *IEEE/ACM Trans. Netw.*, 14(1):41–54, Feb. 2006.

[13] J. Dahl and L. Vandenberghe. CVXOPT: A python package for convex optimization, http://www.abel.ee.ucla.edu/cvxopt, 2006.

[14] D. Donoho and J. Tanner. Observed universality of phase transitions in high-dimensional geometry. *Philosophical Tran. of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367(1906):4273–4293, 2009.

[15] M. Duarte et al. Distributed compressed sensing of jointly sparse signals. In *Conference Record of the 39th Asilomar Conference on Signals, Systems and Computers*, pages 1537 – 1541, 2005.

[16] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, http://www.scipy.org, 2001–.

[17] K. Kanoun, H. Mamaghanian, N. Khaled, and D. Atienza. A real-time compressed sensing-based personal electrocardiogram monitoring system. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2011*, pages 1–6, 2011.

[18] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale l1-regularized least squares. *Selected Topics in Signal Processing, IEEE Journal of*, 1(4):606–617, 2007.

[19] C. Luo et al. Compressive data gathering for large-scale wireless sensor networks. In *Proc. of the 15th annual international conference on Mobile computing and networking (Mobicom)*, pages 145–156, 2009.

[20] S. Mahfoudh and P. Minet. Survey of energy efficient strategies in wireless ad hoc and sensor networks. In *Proc. of the Seventh International Conference on Networking*, pages 1 –7, 2008.

[21] M. Mahmudimanesh, A. Khelil, and N. Suri. Reordering for better compressibility: Efficient spatial sampling in wireless sensor networks. *Proc. of the International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing 2010*, pages 50–57, 2010.

[22] M. Mahmudimanesh, A. Khelil, and N. Suri. Balanced spatio-temporal compressive sensing for multi-hop wireless sensor networks. In *Proceedings of the 9th International Conference on Mobile Ad hoc and Sensor System*, pages 389–397, 2012.

[23] A. O. Marco F. Duarte, Godwin Shen and R. G. Baraniuk. Signal compression in wireless sensor networks. *Philosophical Trans. of the Royal Society*, 370(1958):118–135, 2012.

[24] H. Niederreiter. Recent trends in random number and random vector generation. *Annals of Operations Research*, 31(1):323–345, 1991.

[25] Olfati-Saber et al. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2006.

[26] A. Olshevsky et al. Convergence speed in distributed consensus and averaging. *SIAM J. on Control and Optimization*, 48(1):33–55, 2009.

[27] M. Paselli, F. Petre, O. Rousseaux, G. Meynants, B. Gyselinckx, M. Engels, and L. Benini. A high-performance wireless sensor node for industrial control applications. In *Third International Conference on Systems, ICONS'08*, pages 235–240, 2008.

[28] J. Polastre et al. Telos: enabling ultra-low power wireless research. In *4th Intl. Symp. on Information Processing in Sensor Networks, IPSN 2005*, pages 364–369, 2005.

[29] M. Rabbat et al. Decentralized compression and predistribution via randomized gossiping. In *ACM IPSN'06.*, pages 51–59, 2006.

[30] P. Santi. Topology control in wireless ad hoc and sensor networks. *ACM Comput. Surv.*, 37(2):164–194, June 2005.

[31] K. Sohraby, D. Minoli, and T. Znati. *Wireless Sensor Networks, Technology, Protocols and Applications*. Wiley-Interscience, 2007.

[32] H. O. Tan and I. Körpeoğlu. Power efficient data gathering and aggregation in wireless sensor networks. *SIGMOD Rec.*, 32(4):66–71, Dec. 2003.

[33] R. Verdone, D. Dardari, G. Mazzini, and A. Conti. *Wireless Sensor and Actuator Networks: Technologies, Analysis and Design*. Academic Press, 2008.